International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320-088X

IMPACT FACTOR: 7.056

IJCSMC, Vol. 9, Issue. 11, November 2020, pg.98 – 106

Orthogonal Grey Wolf Optimization Algorithm for Task Scheduling in Cloud Environment

Dr. Md.Yusuf Mulge

Department of Computer Science and Engineering, Nalla Malla Reddy Engineering College, Divya Nagar
Narapally, Ghatkesar(M) Medchal Dist, Telangana State

dryusufmulge@gmail.com

DOI: 10.47760/ijcsmc.2020.v09i11.009

Abstract— Cloud Computing (CC) is being popularly used by small organizations and startups as a business model in distributed computing environment. In cloud computing, it provides three architectures SaaS, the user tasks are organized and executed with suitable resources PaaS and IaaS to deliver the different types of services to the customer. In cloud, the time complexity of task execution is a common problem. In this paper, an efficient Task Scheduling (TS) technique is proposed namely Orthogonal Grey Wolf Optimization (O-GWO) algorithm. The O-GWO algorithm, schedules the tasks on VMs with minimum execution time. Also, it increases the convergence speed and reduces the Degree of Imbalance (DI) of total task schedule across VMs. An experimental analysis shows that, proposed O-GWO algorithm performance is measured with respect to efficient evaluation metrics such as makespan and Degree of Imbalance. The makespan of proposed O-GWO algorithm is achieved to the extent of 38.20 seconds of enhancement compared to the existing Hybrid Particle Swarm Optimization with Simulated Annealing (HPSO-SA).

Keywords— Cloud Computing, Grey Wolf Optimization, Hybrid Particle Swarm optimization, Task Scheduling, Virtual Machines.

I. INTRODUCTION

Over the past decades, Cloud Computing has been popularly used by organizations and academic institutions in the areas of research and development, academics, industrial environment etc.. Cloud computing provides shared pool of resources such as storage, processing, network etc.. through different services like, SaaS (Software as a Service) PaaS (Platform as a Service), and IaaS (Infrastructure as a Service) [2]. In this the cloud users will submit their task to the Cloud Service Provider (CSP), using Internet. The responsibility of the CSP is managing the resources to facilitates the customer. The cloud service provider use a scheduling algorithm to execute incoming tasks and efficiently maintain user computing resources [3, 4]. The ultimate goal of the scheduling algorithm is to allocate the incoming tasks on available processors, increase their utilization, and decrease the total execution time [5]. The existing traditional optimization algorithms are Particle Swarm Algorithm [6], Ant Colony Optimization (ACO) algorithm [7], Chicken Swarm Optimization [8] etc.

The major issues of traditional scheduling algorithms are NP-hard optimization problem for example, combinatorial problem and travelling salesman problem. These problems occurred in VM allocation that may cause delay in TS performance [9]. The traditional ACO method maximizes the task overheads in execution time hence runtime is increased [10]. This research work is majorly concentrated on TS technique in cloud environment namely O-GWO algorithm. The O-GWO algorithm minimized makespan and the DI of total task schedule across VMs. This algorithm selects the multiple processors and reduces the makespan and high convergence speed in minimum time.

The paper is arranged as follows. Section 2 provides a broad survey of many recent papers of different traditional techniques in cloud environment. Section 3 describes the proposed Orthogonal Gray Wolf Optimization algorithm. Further Section 4, is a comparative experimental result of existing and proposed strategy. The conclusion is assessed in Section 5.

II. RELATED WORK

Existing researchers suggested several approaches for task scheduling in cloud. This section, describes the some essential contribution of the existing TS technique.

R. Valarmathi, and T. Sheela, [11] presented a TS technique specifically function of Ranging and Tuning of PSO (RTPSO) algorithm. The traditional PSO algorithm included inertia weight task problem hence, RTPSO method was applied. An important benefit of RTPSO method was minimum run time cost, makespan and improves the convergence speed. Moreover, calculated the resource availability after that allocate the tasks. The RTPSO algorithm consumed high energy utilization in heterogeneous environment.

Yang Li, Xiaofei Liu., *et al.* [12] proposed an efficient multi objective of proposed Genetic Algorithm-based Chaotic Ant Swarm (GA-CAS) algorithm. In cloud GA-CAS method checks the task processing mode and allocates the tasks for available VMs by Markov process and Queuing Theory. The CAS algorithm helps to solve the problem of combinational optimization in search space. Also, GA-CAS performed natural selection and mutation process for solve multi objective issue but operator design cost was maximum.

Y. Moon, H. Yu, J. M. Gil, and J. Lim, [13] presented Slave Ants based ACO (SACO) algorithm for TS in cloud environment. In SACO, the slave ants were avoided the selection of long paths and resolves the problem of global optimization. The SACO algorithm was decreases the preprocessing overheads and makespan. Also, SACO algorithm resolves the problem of NP-hard to increase the utilization of cloud servers. The SACO algorithm shows better performance in homogeneous environment but high energy consumption in heterogeneous environment.

A. S. Kumar, and M. Venkatesan, [14] developed Hybrid GA-PSO (HGPSO) algorithm for TS in cloud. The HGPSO based TS include two major steps such as initially, numerous users' tasks were stored in the queue manager. Second, estimated the task priority and assign the task to the suitable resources. The HGPSO algorithm considers the input from the on-demand queue and selects the appropriate resource for user tasks. The calculation of task priority was depended on the length of the task and memory usage. The significant advantage of the HGPSO algorithm was improved the scalability rate, resource availability and decreases the makespan. The HGPSO algorithm takes more time to complete the task.

C. Thirumalaiselvan, and V. Venkatachalam, [15] presented Rate Based Scheduling (RBS) algorithm of virtual TS in multi-cloud environment. The RBS algorithm provides rate to the tasks and at first maximum rate task was allocated to the VM. In case, the stack was full then, cloud manager check the task priority. The proposed algorithm reduces the delay and energy consumption cost. The significant drawback of RBS algorithm was maximum makespan performance in multi-cloud environment.

III. TASK SCHEDULING OF PROPOSED ORTHOGONAL GRAY WOLF OPTIMIZATION ALGORITHM

Cloud computing model provides the different of on- demand services and it's easy to access the available resources. In cloud, cloud provider has to maintain many users hence, scheduling technique is reduces the cloud provider burden. The TS technique is helps to identify the completed tasks in the VMs. The scheduling algorithm is decreases the task delay, makespan and improve the workload balancing performance. This research work proposed an efficient TS algorithm in cloud environment namely O-GWO algorithm. The O-GWO algorithm majorly concentrates on handling the multiple objectives such as makespan, DI. In order to improve the convergence speed Taguchi optimization technique is combined to the GWO. The proposed architecture is shown in figure.1. The five major steps of the proposed work are described in following section.

Step 1: Initialize the wolf population.

Step 2: The evaluation of fitness values of each wolf is generated by the GWO algorithm and these algorithms generate the candidate response based on the task estimation of VM.

Step 3: Compute all the wolves according to the defined objective (fitness) function in equation 3.

Step 4: In case alpha is not able to select the optimal solution then, proposed O-GWO algorithm helps to trace the best VM for task allocation.

Step 5: Again calculate the position of each wolf and update the value. Finally, obtain the optimal output.

Fig. 1. Proposed method based task scheduling

A. Mathematical task scheduling model

Numerous data centers are involved in cloud and every data centers include set of VMs and performed in parallelly. Let's consider the group of tasks is arranged by cloud users and awaiting tasks are observed. After that, waiting tasks are assigned to the available VMs with less makespan. Consider that number of tasks (cloudlets) is denoted as

$T_i = T_1, T_2, \dots, T_n$ and number of VMs is denoted as V_j

$V_j = V_1, V_2, \dots, V_m$

Assume that a task is scheduled on V_j , the task execution time in virtual machine of cloud is calculated in equation (1).

$$E_{ij} = \frac{T_i}{p(j) \times V_{mips}(j)} \quad (1)$$

Whereas, E_{ij} is indicated as execution time of single task in VM, $p(j)$ is indicated as number of processing elements. $V_{mips}(j)$ is indicated as VM speed in Million Instruction Per Second (MIPS). When more number of VMs are involved to run set of tasks, the total execution time $T_{E(ij)}$ executed on all VMs is calculated in equation (2).

$$T_{E(i,j)} = \sum \left(\frac{T_i}{\dots \times \dots} \right) \quad \forall i = 1, 2, \dots, n, j = 1, 2, \dots, m \quad (2)$$

The mathematical description of cloudlets makespan on different VMs is shown in the eq. (3).

$$T_{E(ij)} = \sum \min \left(\max \left(\sum \left(\frac{T_i}{p(j) \times V_{mips}(j)} \right) \right) \right) \quad (3)$$

B. Orthogonal based Approach

The Orthogonal algorithm is a one kind of optimization method, its ability to solve the complex problem and decreases the execution time of tasks. The method helps to handle the single as well as multi-objective problems. The Orthogonal approach is to construct the Expected Time Compute (ETC) matrix for all the cloudlets. An orthogonal array is mathematically described in the eq. (4).

$$n = \frac{\ln(2^{n-1})}{2k} \quad \text{where } n = 2k \quad (4)$$

Whereas, -1 is signifies the column number in two levels of orthogonal array, $n = 2k$ is indicated as number of iterations corresponding to the n rows and columns. Number of required levels for each factor is Z . the variable k is indicated as positive integer. In this algorithm any column pairs and combination pairs are performed same number of times in every level. Taguchi method construct the ETC matrix of all tasks on VMs after that O-GWO algorithm is tracing the optimal solution and mapping the VMs with minimum ETC. The O-GWO algorithm helps to decrease the makespan of total scheduled tasks and it's described in the following sections.

C. Grey Wolf Optimization

The GWO is a swarm intelligence algorithm and major responsibility is the leadership hierarchy and bio-inspired natures. The fitness values are calculated by three major wolves such as alpha, beta and gamma. These three wolves are guided to the other wolves and alpha wolf's fitness value is considered as best value. The major activities of GWO algorithm is encircling, searching and attacking the prey.

Let's consider that $w_1 = w_2 = \dots = w_n$ is indicated as position vectors in the search space and n is represented as dimension of the vectors. The fitness function is helps to estimate the wolves position. The wolves solutions are depends on fitness values. Also, best solutions are classified as first solution is indicted as α , second solution is β and third solution is γ . In searching process, these wolves' position values are updated. At first,

wolves positions are initialized and co-efficient vectors are $\vec{A}$ and $\vec{C}$ described in eq.(5) and (6).

$$\vec{A} = 2\vec{a} \cdot \vec{r1} - \vec{a} \quad (5)$$

$$\vec{C} = 2\vec{r2} \quad (6)$$

Whereas, vector $\vec{A}$ considers the random values in the range of $[-a, a]$, and vector $\vec{C}$ is in the range of $[-a, a]$, Its helps to choose the optimal solution in the search space. After initialization of each wolves, fitness values are Degree third such as α , β and δ respectively. The mathematical calculation is shown in the equation (7), (8), and (9).

$$\vec{D}_\alpha = |\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}| \cdot |\vec{D}|, \vec{D}_\beta = |\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}|, \vec{D}_\delta = |\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}| \quad (7)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot (\vec{D}_\alpha), \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot (\vec{D}_\beta), \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot (\vec{D}_\delta) \quad (8)$$

$$\vec{X}_{(t+1)} = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (9)$$

In each iteration of the algorithm, the wolf's position update depends on the position of wolves such as α , β , and δ .

In addition, values of vectors $\vec{A}$, $\vec{C}$, and $\vec{a}$ are updated in the current position of the wolves. As the velocity of wolf points shows a near-optimal solution, sets of candidate velocity vectors α , β , and δ are indicated as $V_{set1}(t)$, $V_{set2}(t)$, and $V_{set3}(t)$ based on Taguchi design are generated as followed in equation (10)

$$\vec{V}_{ok,d}(t) = \begin{cases} V_{set1,d}(t) = w_1 \cdot \vec{V}_{ok,d}(t-1) + c_1 \cdot r_1 \cdot (\vec{X}_{best,d}(t-1) - \vec{X}_{ok,d}(t-1)) \\ V_{set2,d}(t) = w_2 \cdot \vec{V}_{ok,d}(t-1) + (c_2 \cdot r_2 \cdot (\vec{X}_{best,d}(t-1) - \vec{X}_{ok,d}(t-1))) \\ V_{set3,d}(t) = w_3 \cdot \vec{V}_{ok,d}(t-1) + (c_3 \cdot r_3 \cdot (\vec{X}_{best,d}(t-1) - \vec{X}_{ok,d}(t-1))) \end{cases} \quad (10)$$

Whereas, $\vec{V}_{ok,d}(t)$ indicates the velocity sets of three wolves candidates such as α , β , and δ and u is the dimension of the solution space. $\vec{X}_{best,d}$ is represents the global best position attained by the wolf. The variable w_1, w_2 and w_3 are indicates the controlled factors of α, β , and δ candidates. The variable r_1, r_2 and r_3 are the uniform random number in the range of $[0,1]$ and c_1, c_2, c_3 are the constant value that indicates the acceleration. $\vec{X}_{k,d}$ indicated as position of the wolf and k is the number of iteration. The OTB approach is incorporated with GWO algorithm for tracing the optimal solution with maximum convergence speed and minimum task execution time.

IV. EXPERIMENTAL RESULT AND DISCUSSION

For experimental simulation, CloudSim 3.0.3 PlanetLab workload was employed on PC with 3.2 GHz with i5 processor. The O-GWO algorithm performance was compared with the traditional TS techniques such as HPSO- SA and O-Cat Swarm Optimization (O-CSO).

A. *Evaluation Metrics* The TS performance of O-GWO method is measured using two significant evaluation parameters such as makespan and DI

❖ Degree of Imbalance

The VM overloaded tasks, resource usage of RAM, MIPS and bandwidth and cloudlet traffic creates the DI in cloud environment. The DI is used to measure imbalance. The DI is used to measure imbalance among VMs to spread workload evenly for VM and it's described in equation (11),

$$DI = \frac{T_{\max} - T_{\min}}{T_{\text{avg}}} \quad (11)$$

Whereas, maximum time period is indicated as $T_{\max}$ and minimum time period is represented as $T_{\min}$. The average time period of single task execution time is depicted as T_{avg} .

❖ Makespan

It's defined as the task completion time or execution time of the assigned task in VMs. The mathematical equation of makespan is shown in the equation (12)

$$T_E = \left(\frac{W_i}{P_i} \right) * X_{ij} \quad (12)$$

TABLE I: PERFORMANCE OF MAKESPAN (SEC) OBTAINED WITH 10 VMS

No. Task	HPSO-SA [16]			O-CSO [16]			Proposed O-GWO		
	Best	Worst	Avg	Best	Worst	Avg	Best	Worst	Avg
10	6.99	49.05	25.81	3.96	11.64	10.35	2.12	7.41	8.54
20	22.72	131.01	53.19	16.01	76.40	49.52	12.45	41.52	35.47
30	49.59	114.12	74.61	36.01	134.56	69.91	25.87	95.47	57.24
40	113.00	367.91	86.16	85.49	218.91	138.62	50.14	110.41	60.11
50	119.69	536.23	248.60	111.95	312.21	231.63	85.68	212.89	114.36
60	216.94	600.68	79.09	138.61	416.26	210.82	101.87	282.47	62.45

Fig. 2. Performance of Average Makespan

The figure.2 indicates the graphical representation of average case task execution time on VMs. In Average case, the makespan performance of HPSO-SA algorithm achieved maximum 248.60 sec in 50 tasks and minimum 25.81 sec in 10 tasks. Here, clearly observed that, number of tasks are increased then, task completion time also increased in all the cases. Similarly, O-CSO algorithm achieved performance of makespan minimum 10.35 sec in 10 number of tasks and maximum 231.63 sec in 50 number of tasks. Finally, the proposed O-GWO algorithm achieve Performance of makespan minimum 8.54sec in 10 tasks and maximum 114.36 sec in 50 tasks.

Table II: Performance of DI for 10 VMS

No. Task	HPSO-SA [16]			O-CSO [16]			Proposed O-GWO		
	Best	Worst	Avg	Best	Worst	Avg	Best	Worst	Avg
10	0.51	9.17	2.42	0.27	10.41	5.63	0.4	7.12	1.36
20	0.35	6.74	4.85	0.11	7.94	3.59	0.8	3.78	2.39
30	0.12	21.26	8.75	0.22	15.15	14.38	0.11	14.75	6.54
40	0.23	23.8	15.60	0.10	48.79	25.53	0.8	18.63	13.78
50	0.24	44.22	22.28	0.22	12.73	16.08	0.18	10.23	15.69
60	0.26	69.09	38.86	0.10	25.93	21.53	0.9	21.36	19.99

The table.2 shows the performance of DI with respect to different set of tasks of 10 VMs. If number of task increased then workloads are difficult to balance in the system. An existing HPSO-SA algorithm shows the maximum DI compare to the O-CSO. The proposed O-GWO algorithm shows the minimum DI. If the number of task increased then, it's able to handle the workloads. The graphical representation is shown in figure.3.

Fig.3. Performance of Average Degree of Imbalance

The figure.3 indicates the performance of DI with respect to different set of tasks. Here, clearly observed that number of tasks is increased then DI values show more variations. In average case, the traditional HPSO-SA algorithm achieved maximum DI is 38.86% and minimum DI is 2.42%. Similarly, O-CSO algorithm achieved maximum 25.53% of DI and minimum 3.59% of DI. Finally, proposed O-GWO algorithm achieved minimum 1.36% of DI and maximum 19.99% of DI.

Experimental outcomes which represent the different sets of tasks are not impacts on the proposed OTB-GWO. Compare to the existing O-CSO and HPSO-SA, the proposed O-GWO shown better results in both make span and DI. Moreover, O-GWO algorithm well balance all the sets of tasks compared to the existing algorithms.

V. CONCLUSION

Cloud computing is the emerging technology that provides the various on-demand services to the all cloud users in any part of the world. In cloud, TS is the kind of issue by considering different factors such as task completion time, utilization of resources, power utilization and etc. In this paper, an efficient multi-objective TS technique is proposed namely O-GWO algorithm. The proposed algorithm efficiently schedules the tasks into appropriate VMs hence; significantly reduce the makespan and DI. An experimental analysis is done on CloudSim tool taking 10 virtual machines. It has shown that the proposed O-GWO algorithm task scheduling performance is measured using makespan and DI in terms of three cases (best, worst and average). The proposed O-GWO achieved approximately 38.20sec and 4.44% of enhancement in terms of makespan and DI respectively. Compare to the traditional HPSO-SA and O-CSO, the proposed O-GWO algorithm shown better results. In future, an efficient hybrid TS technique is applied on heterogeneous environment.

REFERENCES

1. Z. Zhang, "Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: Overview and challenges" IEEE Communication /surveys and Tutorials 8, vol. 4, January, 2006, pp. 24-37
2. K. Fall, "A Delay Tolerant Network Architecture for Challenged Internets" Proc. Of Annual Conf. Of the Special Interest Group on Data Communication (ACM SIGCOMM'03), pp. 27-34, Aug. 2003.
3. Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., Weiss, H.: RFC 4838, Delay-Tolerant Networking Architecture. IRTF DTN Research Group (2007).
4. Eyuphan Bulut, "Opportunistic Routing Algorithms in Delay Tolerant Networks", A Thesis of DOCTOR OF PHILOSOPHY, submitted on May 2011.
5. S. Jain, K. Fall, and R. Patra, "Routing in Delay Tolerant Networks", In Proc. ACM SIGCOMM, 2004.
6. A. Demers, D. Greene, C. Houser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swineheart, and D. Terry, "Epidemic Algorithms for Replicated Database Maintenance", ACM SIGPOS Operating system Review, V.22, N.1, Jan. 1988.
7. O. Gnawali, M. Polyakov, P. Bose, R. Govindan, : Data centric, position-based routing in space networks", In Proc. 26Th IEEE Aerospace Conference, pp. 1322-1334, 2005.
8. Merugu S., Ammar M., Zegura E., Routing in Space and Time in Networks with Predictable Mobility", Tech. Report, GIT-CC-04-7, Georgia Institute of Tech.,2004. Available at <http://smartech.gatech.edu/handle/1853/6492>.
9. R. Handorean, C. Gill, G.-C. Roman, "Accommodating transient connectivity in ad hoc and mobile settings", LNCS, V. 3001/2004, Springer, pp. 305-322, Jan. 2004.
10. I. Cardei, C. Liu, J. Wu, "Routing in Wireless Networks with Intermittent Connectivity", Encyclopedia of Wireless and Mobile Communications, B. Furht (ed.), CRC Press, Taylor and Francis Group, 2007.
11. B. Burns, O. Brock, and B. N. Levine, "MV routing and capacity building in disruption tolerant networks", In proc. IEEE INFOCOM, pp. 398-408, March 2005.
12. W. Zhao, M. Ammar, E. Zegura, "A message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks", In. Proc. Of 5th MobiHoc, pp. 113-117,2004.
13. A. Vahdatand, D. Becker, "Epidemic Routing for Partially Connected Ad Hoc Networks", Duke Technical Report, CS-2000-06, July 2000. available at issg.cs.duke.edu/epidemic/epidemic.pdf.
14. Evan P. C. Jones and Paul A. S. Ward, "Routing Strategies for Delay-Tolerant Networks", 2006. available at citeseerx.ist.psu.edu
15. T. Spyropoulos, K. Psounis, C. Raghavendra, "Spray-and-Wait: Efficient routing scheme for intermittently connected mobile networks", in ACM SIGCOMM Workshop on Delay Tolerant Networking (WDTN), 2005.
16. A. Lindgren, A. Doria, O. Schelen, "Probabilistic routing in intermittently connected networks", ACM SIGMobile Computing and Communication Review, V.7, N.1, July 2003.
17. Jeremie Leguay, Timur Friedman, Vania Conan, "Evaluating Mobility Pattern Space Routing for DTNs", In Proc. IEEE INFOCOM , pp. 1-10, April 2006.
18. Jian Zhang, Yuanzhu Peter Chen, Ivan Marsic, "Networking Coding via Opportunistic Forwarding in Wireless Mesh Networks", In Proc. WCNC, pp. 1775-1780, 2008.